Aprendizagem 2021/22
Homework III – Group 072

I. Pen-and-paper

1)

a)

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\tanh'(x) = \frac{(e^x - e^{-x})'(e^x + e^{-x}) - (e^x - e^{-x})(e^x + e^{-x})'}{(e^x + e^{-x})^2} = \frac{(e^x + e^{-x})^2 - (e^x - e^{-x})^2}{(e^x + e^{-x})^2} = 1 - \left(\frac{e^x - e^{-x}}{e^x + e^{-x}}\right)^2 = 1 - \tanh^2(x)$$

Considerando os pesos e os bias para fazer uma stochastic descent update:

$$W^{[1]} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad W^{[2]} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad b^{[2]} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad W^{[3]} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad b^{[3]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$z = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad X^{[0]} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

1º) Fazemos uma forward propagation:

$$T^{[1]} = W^{[1]} \cdot X^{[0]} + b^{[1]} = \begin{bmatrix} 6 \\ 1 \\ 6 \end{bmatrix} \qquad X^{[1]} = \tanh\left(\begin{bmatrix} 6 \\ 1 \\ 6 \end{bmatrix}\right) = \begin{bmatrix} 0.9999877 \\ 0.761594 \\ 0.9999877 \end{bmatrix}$$

$$T^{[2]} = W^{[2]} \cdot X^{[1]} + b^{[2]} = \begin{bmatrix} 3.7615694 \\ 3.7615694 \end{bmatrix} \qquad X^{[2]} = \tanh\left(\begin{bmatrix} 3.7615694 \\ 3.7615694 \end{bmatrix}\right) = \begin{bmatrix} 0.99891972 \\ 0.99891972 \end{bmatrix}$$

$$T^{[3]} = W^{[3]} \cdot X^{[2]} + b^{[3]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad X^{[3]} = \tanh\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

2º) Fazemos uma backward propagation, em que a squared error measure é:

$$E(z, x^{[L]}) = \frac{1}{2}\left(x^{[L]} - z\right)^2 \qquad \frac{\partial E}{\partial X^{[L]}}(z, X^{[L]}) = X^{[L]} - z$$

$$\frac{\partial X^{[L]}}{\partial T^{[L]}}\left(T^{[L]}\right) = 1 - \tanh^2\left(T^{[L]}\right) \qquad \frac{\partial T^{[L]}}{\partial W^{[L]}}\left(W^{[L]}, b^{[L]}, X^{[L-1]}\right) = X^{[L-1]}$$

$$\frac{\partial T^{[L]}}{\partial b^{[L]}}\left(W^{[L]}, T^{[L]}, X^{[L-1]}\right) = 1 \qquad \frac{\partial T^{[L]}}{\partial X^{[L-1]}}\left(W^{[L]}, b^{[L]}, X^{[L-1]}\right) = W^{[L]}$$

Começamos a recursão descobrindo o delta da última chamada

$$\delta^{[3]} = \frac{\partial E}{\partial X^{[3]}} \circ \frac{\partial X^{[3]}}{\partial T^{[3]}} = \left(X^{[3]} - z\right)^T \circ \left(1 - \tanh^2(\mathrm{T}^{[3]})\right) = \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) \circ \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} \tanh^2(0) \\ \tanh^2(0) \end{bmatrix}\right) = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\delta^{[2]} = \frac{\partial T^{[3]}}{\partial X^{[2]}} \cdot \delta^{[3]} \circ \frac{\partial X^{[2]}}{\partial T^{[2]}} = \left(W^{[3]}\right)^T \cdot \delta^{[3]} \circ \left(1 - \tanh^2\left(T^{[2]}\right)\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\delta^{[1]} = \frac{\partial T^{[2]}}{\partial X^{[1]}} \cdot \delta^{[2]} \circ \frac{\partial X^{[1]}}{\partial T^{[1]}} = \left(W^{[2]}\right)^T \cdot \delta^{[2]} \circ \left(1 - \tanh^2\left(T^{[1]}\right)\right) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

3º) Atualiza-se os pesos

$$\frac{\partial E}{\partial W^{[1]}} = \delta^{[1]} \cdot \frac{\partial T^{[1]}}{\partial W^{[1]}}^T = \delta^{[1]} \cdot \left(W^{[0]}\right)^T = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$W^{[1]} = W^{[1]} - \eta\frac{\partial E}{\partial W^{[1]}} = \begin{bmatrix}1&1&1&1&1\\0&0&0&0&0\\1&1&1&1&1\end{bmatrix} - 0.1\begin{bmatrix}0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix} = \begin{bmatrix}1&1&1&1&1\\0&0&0&0&0\\1&1&1&1&1\end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]}\cdot\frac{\partial {T^{[1]}}^T}{\partial b^{[1]}} = \delta^{[1]} = \begin{bmatrix}0\\0\\0\end{bmatrix} \qquad b^{[1]} = b^{[1]} - \eta\frac{\partial E}{\partial b^{[1]}} = \begin{bmatrix}1\\1\\1\end{bmatrix} - 0.1\begin{bmatrix}0\\0\\0\end{bmatrix} = \begin{bmatrix}1\\1\\1\end{bmatrix}$$

$$\frac{\partial E}{\partial W^{[2]}} = \delta^{[2]}\cdot\frac{\partial {T^{[2]}}^T}{\partial W^{[2]}} = \delta^{[2]}\cdot\left(W^{[1]}\right)^T = \begin{bmatrix}0&0&0\\0&0&0\end{bmatrix}$$

$$W^{[2]} = W^{[2]} - \eta\frac{\partial E}{\partial W^{[2]}} = \begin{bmatrix}1&1&1\\1&1&1\end{bmatrix} - 0.1\begin{bmatrix}0&0&0\\0&0&0\end{bmatrix} = \begin{bmatrix}1&1&1\\1&1&1\end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]}\cdot\frac{\partial {T^{[2]}}^T}{\partial b^{[2]}} = \delta^{[2]} = \begin{bmatrix}0\\0\end{bmatrix} \qquad b^{[2]} = b^{[2]} - \eta\frac{\partial E}{\partial b^{[2]}} = \begin{bmatrix}1\\1\end{bmatrix} - 0.1\begin{bmatrix}0\\0\end{bmatrix} = \begin{bmatrix}1\\1\end{bmatrix}$$

$$\frac{\partial E}{\partial W^{[3]}} = \delta^{[3]}\cdot\frac{\partial {T^{[3]}}^T}{\partial W^{[3]}} = \delta^{[3]}\cdot\left(W^{[2]}\right)^T = \begin{bmatrix}-0.99891972&-0.99891972\\0.99891972&0.99891972\end{bmatrix}$$

$$W^{[3]} = W^{[3]} - \eta\frac{\partial E}{\partial W^{[3]}} = \begin{bmatrix}0&0\\0&0\end{bmatrix} - 0.1\begin{bmatrix}-0.99891972&-0.99891972\\0.99891972&0.99891972\end{bmatrix} = \begin{bmatrix}0.099891972&0.099891972\\-0.99891972&-0.99891972\end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[3]}} = \delta^{[3]}\cdot\frac{\partial {T^{[3]}}^T}{\partial b^{[3]}} = \delta^{[3]} = \begin{bmatrix}-1\\1\end{bmatrix} \qquad b^{[3]} = b^{[3]} - \eta\frac{\partial E}{\partial b^{[3]}} = \begin{bmatrix}0\\0\end{bmatrix} - 0.1\begin{bmatrix}-1\\1\end{bmatrix} = \begin{bmatrix}0.1\\-0.1\end{bmatrix}$$

b) $\text{softmax}([\mathbf{z_1}\quad z_2\quad \dots\quad z_d]^T) = [x_1\quad x_2\quad \dots\quad x_d]^T \qquad z = target(t)$

Se i=j

$$\frac{\partial x_i}{\partial z_j} = \frac{\partial}{\partial z_i}\frac{e^{z_i}}{\sum_{k=1}^d e^{z_k}} = \frac{\left(\frac{\partial}{\partial z_j}e^{z_i}\right)\left(\sum_{k=1}^d e^{z_k}\right) - e^{z_i}\left(\frac{\partial}{\partial z_j}\sum_{k=1}^d e^{z_k}\right)}{\left(\sum_{k=1}^d e^{z_k}\right)^2} = \frac{e^{z_i}\left(\sum_{k=1}^d e^{z_k}\right) - e^{z_i}e^{z_j}}{\left(\sum_{k=1}^d e^{z_k}\right)^2}$$

$$= \frac{e^{z_i}}{\sum_{k=1}^d e^{z_k}}\left(1 - \frac{e^{z_j}}{\sum_{k=1}^d e^{z_k}}\right) = x_i\left(1-x_j\right) = x_i\left(1-x_i\right)$$

Se i≠j

$$\frac{\partial x_i}{\partial z_j} = \frac{\partial}{\partial z_j}\frac{e^{z_i}}{\sum_{k=1}^d e^{z_k}} = e^{z_i}\frac{\partial}{\partial z_j}\frac{1}{\sum_{k=1}^d e^{z_k}} = e^{z_i}\frac{\partial\left(\frac{1}{\sum_{k=1}^d e^{z_k}}\right)\partial\left(\sum_{k=1}^d e^{z_k}\right)}{\partial\left(\sum_{k=1}^d e^{z_k}\right)\partial z_j} = e^{z_i}\left(-\frac{1}{\left(\sum_{k=1}^d e^{z_k}\right)^2}\right)e^{z_j}$$

$$= -\frac{e^{z_i}}{\sum_{k=1}^d e^{z_k}}\frac{e^{z_j}}{\sum_{k=1}^d e^{z_k}} = -x_i x_j$$

$$\delta_i^{[3]} = \frac{\partial E\left(t, X^{[3]}\right)}{\partial z_i} = \frac{\partial}{\partial z_i}\left(-\sum_{k=1}^d t_k log\left(x_k^{[3]}\right)\right) = -\sum_{k=1}^d t_k\frac{1}{x_k^{[3]}}\frac{\partial x_k^{[3]}}{\partial z_i} = -\sum_{k=i} t_k\frac{1}{x_k^{[3]}}\frac{\partial x_k^{[3]}}{\partial z_i} - \sum_{k\neq i} t_k\frac{1}{x_k^{[3]}}\frac{\partial x_k^{[3]}}{\partial z_i} =$$

$$= -\sum_{k=i}\frac{t_k x_i^{[3]}\left(1-x_i^{[3]}\right)}{x_i^{[3]}} - \sum_{k\neq i}\frac{-t_k x_k^{[3]}x_i^{[3]}}{x_i^{[3]}} = -t_i\left(1-x_i^{[3]}\right) + \sum_{k\neq i} t_k x_i^{[3]} = -t_i + x_i^{[3]}\left(t_i + \sum_{k\neq i} t_k\right)$$

$$= -t_i + x_i^{[3]}\left(\sum_{k=1}^d t_k\right) = x_i^{[3]} - t_i$$

$$\frac{\partial X^{[L]}}{\partial z^{[L]}}\left(z^{[L]}\right) = 1 - \tanh^2\left(z^{[L]}\right) \qquad \frac{\partial z^{[L]}}{\partial W^{[L]}}\left(W^{[L]}, b^{[L]}, X^{[L-1]}\right) = X^{[L-1]}$$

$$\frac{\partial z^{[L]}}{\partial b^{[L]}}\left(W^{[L]}, z^{[L]}, X^{[L-1]}\right) = 1 \qquad \frac{\partial z^{[L]}}{\partial X^{[L-1]}}\left(W^{[L]}, b^{[L]}, X^{[L-1]}\right) = W^{[L]}$$

$$W^{[1]} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad W^{[2]} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad b^{[2]} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad W^{[3]} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad b^{[3]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$z = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad X^{[0]} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$z^{[1]} = W^{[1]} \cdot X^{[0]} + b^{[1]} = \begin{bmatrix} 6 \\ 1 \\ 6 \end{bmatrix} \qquad X^{[1]} = \tanh\left(\begin{bmatrix} 6 \\ 1 \\ 6 \end{bmatrix}\right) = \begin{bmatrix} 0.9999877 \\ 0.761594 \\ 0.9999877 \end{bmatrix}$$

$$z^{[2]} = W^{[2]} \cdot X^{[1]} + b^{[2]} = \begin{bmatrix} 3.7615694 \\ 3.7615694 \end{bmatrix} \qquad X^{[2]} = \tanh\left(\begin{bmatrix} 3.7615694 \\ 3.7615694 \end{bmatrix}\right) = \begin{bmatrix} 0.99891972 \\ 0.99891972 \end{bmatrix}$$

$$z^{[3]} = W^{[3]} \cdot X^{[2]} + b^{[3]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad X^{[3]} = \mathrm{softmax}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\delta^{[3]} = \begin{pmatrix} \delta_1^{[3]} \\ \delta_2^{[3]} \\ \delta_3^{[3]} \end{pmatrix} = \begin{pmatrix} x_1^{[3]} - t_1 \\ x_2^{[3]} - t_2 \\ x_3^{[3]} - t_3 \end{pmatrix} = \left(x^{[3]} - t\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

$$\delta^{[2]} = \frac{\partial z^{[3]}}{\partial X^{[2]}} \cdot \delta^{[3]} \circ \frac{\partial X^{[2]}}{\partial z^{[2]}} = \left(W^{[3]}\right)^T \cdot \ \delta^{[3]} \circ \left(1 - \tanh^2\left(z^{[2]}\right)\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\delta^{[1]} = \frac{\partial z^{[2]}}{\partial X^{[1]}} \cdot \delta^{[2]} \circ \frac{\partial X^{[1]}}{\partial z^{[1]}} = \left(W^{[2]}\right)^T \cdot \ \delta^{[2]} \circ \left(1 - \tanh^2\left(z^{[1]}\right)\right) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\frac{\partial E}{\partial W^{[1]}} = \delta^{[1]} \cdot \frac{\partial T^{[1]}}{\partial W^{[1]}}^T = \delta^{[1]} \cdot \left(W^{[0]}\right)^T = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$W^{[1]} = W^{[1]} - \eta \frac{\partial E}{\partial W^{[1]}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} - 0.1 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[1]}} = \delta^{[1]} \cdot \frac{\partial z^{[1]}}{\partial b^{[1]}}^T = \delta^{[1]} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad b^{[1]} = b^{[1]} - \eta \frac{\partial E}{\partial b^{[1]}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - 0.1 \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\frac{\partial E}{\partial W^{[2]}} = \delta^{[2]} \cdot \frac{\partial z^{[2]}}{\partial W^{[2]}}^T = \delta^{[2]} \cdot \left(W^{[1]}\right)^T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$W^{[2]} = W^{[2]} - \eta \frac{\partial E}{\partial W^{[2]}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} - 0.1 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[2]}} = \delta^{[2]} \cdot \frac{\partial z^{[2]}}{\partial b^{[2]}}^T = \delta^{[2]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad b^{[2]} = b^{[2]} - \eta \frac{\partial E}{\partial b^{[2]}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} - 0.1 \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\frac{\partial E}{\partial W^{[3]}} = \delta^{[3]} \cdot \frac{\partial z^{[3]}}{\partial W^{[3]}}^T = \delta^{[3]} \cdot \left(W^{[2]}\right)^T = \begin{bmatrix} -0.99891972 & -0.99891972 \\ 0 & 0 \end{bmatrix}$$

$$W^{[3]} = W^{[3]} - \eta \frac{\partial E}{\partial W^{[3]}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} - 0.1 \begin{bmatrix} -0.99891972 & -0.99891972 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0.099891972 & 0.099891972 \\ 0 & 0 \end{bmatrix}$$

$$\frac{\partial E}{\partial b^{[3]}} = \delta^{[3]} \cdot \frac{\partial z^{[3]}}{\partial b^{[3]}}^T = \delta^{[3]} = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \qquad b^{[3]} = b^{[3]} - \eta \frac{\partial E}{\partial b^{[3]}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 0.1 \begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0 \end{bmatrix}$$

Aprendizagem 2021/22
Homework III – Group 072

II. Programming and critical analysis

2)

Chegou-se a conclusão de que, em cada iteração do algoritmo os valores das métricas associadas a matriz de confusão alteram, consideramos duas das iterações para o estudo.

Numa primeira iteração com os mesmo valores de max_iter(max_iter = 1120) para o classificador com early_stopping=true e o contrário, concluímos que a log loss é menor para early_stopping=false o que traduz-se numa melhor previsão(para um qualquer problema, uma menor valor de log-loss traduz-se em melhores previsões).

Classifier	Brier loss	Log loss	F1	Accuracy
mlp early stoping=true	0.184366	0.549684	0.000000	0.647059
mlp early stopping= false	0.155255	0.495760	0.675676	0.823529

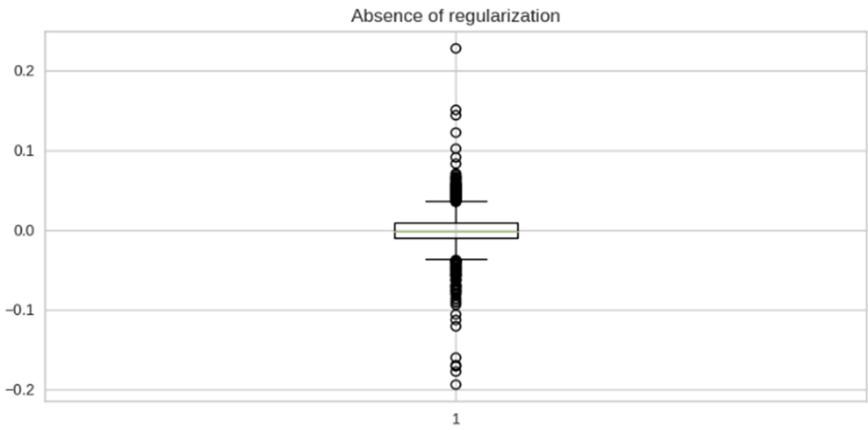
Numa segunda iteração, atribuímos o mesmo valor de max_iter=2000, para dois classificadores (mlp com early_stopping=true e mlp com early_stopping=false), neste obtivemos o seguinte:

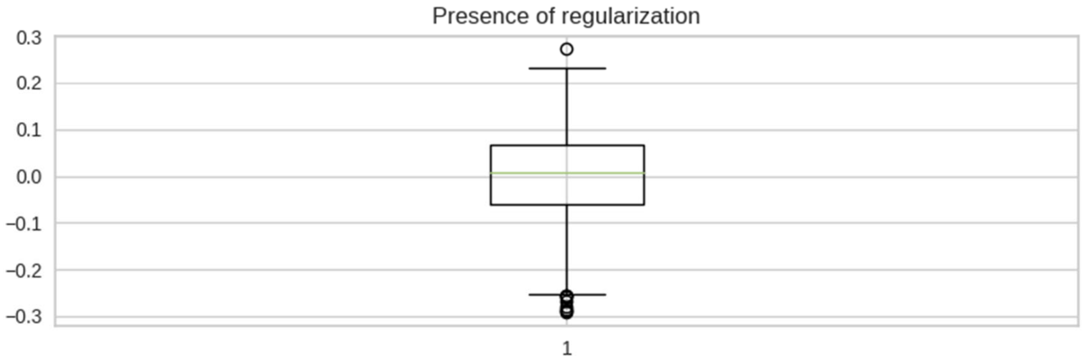
Classifier	Brier loss	Log loss	F1	Accuracy
mlp early stoping=true	0.224760	0.662867	0.682927	0.808824
mlp early stopping= false	0.382476	0.977922	0.521739	0.352941

Como se pode verificar o modelo teve melhores métricas na presença do early stopping, isto é, parou de treinar o *dataset* a partir dum certo mínimo (minimizar o *overfit*), ao passo que com o early stopping= false, o modelo continuou a treinar o *dataset,* o que resultou posteriormente em piores resultados com o dataset de teste.

Aprendizagem 2021/22
Homework III – Group 072

3)





De forma a minimizar o erro observado do *regressor* MLP, podemos adotar um conjunto de estratégias, nomeadamente: Aumentar o conjunto de dados a testar e treinar, alterar a função de ativação, sendo que esta última tem um impacto enorme na performance da rede neuronal, geralmente usa-se a mesma função de ativação para todos as *hiddens layers* e uma outra diferente para o output layer, pode-se neste caso, utilizar uma função de ativação linear para a *hidden layer.*

III. APPENDIX

2.
```
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import (
    plot_confusion_matrix,
    classification_report,
    accuracy_score,
    log_loss,
    f1_score,
    log_loss,
    brier_score_loss
)


import matplotlib.pyplot as plt
```

Aprendizagem 2021/22

Homework III – Group 072

```
import pandas as pd
import numpy as np
from collections import defaultdict
import dataframe_image as dfi

#different attributes plus the class
col =
['Clump_Thickness','Cell_Size_Uniformity','Cell_Shape_Uniformity','Marginal_Adhesion','Single_Epi_Ce
ll_Size',
'Bare_Nuclei','Bland_Chromatin','Normal_Nucleoli','Mitoses','Class']

#loads the data including the target values
data = pd.read_csv('result-breast.csv',usecols=col)
#loads the data and only the data
dataset = pd.read_csv('result-breast.csv',usecols=col[:9])

#creates a data frame
df = pd.DataFrame(dataset)

#obtains an array with the target values
y = np.array([value for value in data['Class']])

#obtains an array with the attributes
X = df.values


# splits into train and test datasets
skf = StratifiedKFold(n_splits=5,random_state= 0,shuffle=True)
for train_index, test_index in skf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]



# scales the data in order to prevent a high convergence time (SGD)
scaler = StandardScaler()
scaler.fit(X_train)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)


# creates two different classifiers, one of them with early_stopping set
# to true and the other with early_stopping as default, false
mlp_true = MLPClassifier(hidden_layer_sizes= (3,2),max_iter=2000,early_stopping=True)
mlp_false = MLPClassifier(hidden_layer_sizes=(3,2),max_iter=2000)

# classifiers list
mlp_list = [
    (mlp_true, "mlp early stoping=true"),
    (mlp_true,"mlp early stopping= false")
    ]

# used to store metrics
scores = defaultdict(list)


for i, (mlp, name) in enumerate(mlp_list):
    #fits the training data to our classification models
    mlp.fit(X_train, y_train)
    y_prob = mlp.predict_proba(X_test)
    y_pred = mlp.predict(X_test)
    scores["Classifier"].append(name)
```

```
    for metric in [brier_score_loss, log_loss]:
        score_name = metric.__name__.replace("_", " ").replace("score", "").capitalize()
        scores[score_name].append(metric(y_test, y_prob[:, 1]))

    for metric in [f1_score,accuracy_score]:
        score_name = metric.__name__.replace("_", " ").replace("score", "").capitalize()
        scores[score_name].append(metric(y_test, y_pred))


    score_df = pd.DataFrame(scores).set_index("Classifier")
    score_df.round(decimals=3)


df_styled = score_df.style.background_gradient()
dfi.export(df_styled,"results-ex2.png")
```

3.

```
from sklearn.neural_network import MLPRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import KFold
from yellowbrick.regressor import ResidualsPlot
from sklearn.pipeline import make_pipeline
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import dataframe_image as dfi

#different attributes
col = ['theta1','theta2','theta3',
'theta4','theta5','theta6','theta7','theta8','y'
]

#loads the data
data = pd.read_csv('kin8nm.csv',usecols=col)

#creates a data frame
df = pd.DataFrame(data)

#obtains an array with the target values
y = np.array([value for value in data['y']])

#obtains an array with the attributes
X = df.values



mlp_reg = make_pipeline(
    StandardScaler(),
    MLPRegressor(hidden_layer_sizes=(3,2), random_state=0,alpha=10)
)

# splits into train and test datasets
kf = KFold(n_splits=5,random_state= 0,shuffle=True)
for train_index, test_index in kf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    mlp_reg.fit(X_train, y_train)



y_pred = mlp_reg.predict(X_test)
```

```
#computes the residues
residues = y_test - y_pred

plt.boxplot(residues)
plt.title("Presence of regularization")
plt.show()
```

END